Getting Started with Quarkus

Michael P. Redlich

Who's Mike?

- Bachelor of Science, Computer Science
- "Petrochemical Research Organization"
- Java Queue News Editor, InfoQ
- Leadership Council, Jakarta EE Ambassadors
- Amateur Computer Group of New Jersey

RUTGERS
THE STATE UNIVERSITY OF NEW JERSEY

Objectives

- What is Quarkus?
- Why Quarkus?
- Get Started Building an Application
  - Live Demo (yea!)
- Quarkus Resources

What is Quarkus?

"A Kubernetes Native Java stack tailored for OpenJDK HotSpot and GraalVM, crafted from the best-of-breed Java libraries and standards."

Quarkus website: https://quarkus.io

What is Quarkus?

- Supersonic Subatomic Java!
- A cohesive, fun-to-use, full-stack framework
- A portmanteau of:
  - quark
  - us
- Latest version: 1.1.1

What is OpenJDK HotSpot?

- The default Java Virtual Machine (JVM) for the JDK since Java 1.3
- Continually analyzes a program's performance for hot spots
- Written in C++

What is GraalVM?

"A universal virtual machine for running applications in JavaScript, Python, Ruby, R, JVM-based and LLVM-based languages."

GraalVM website: https://graalvm.org

What is GraalVM?

- Builds a native image (standalone executable) of an application
- Written in Java
- Latest version: 19.3.0.2

Why Quarkus?

- Modern applications:
  - Cloud, mobile and IoT
  - Containers, orchestration, microservices, reactive, FaaS, 12-factor, cloud-native
- Re-think how Java can best be utilized to address these new deployment environments and application architectures

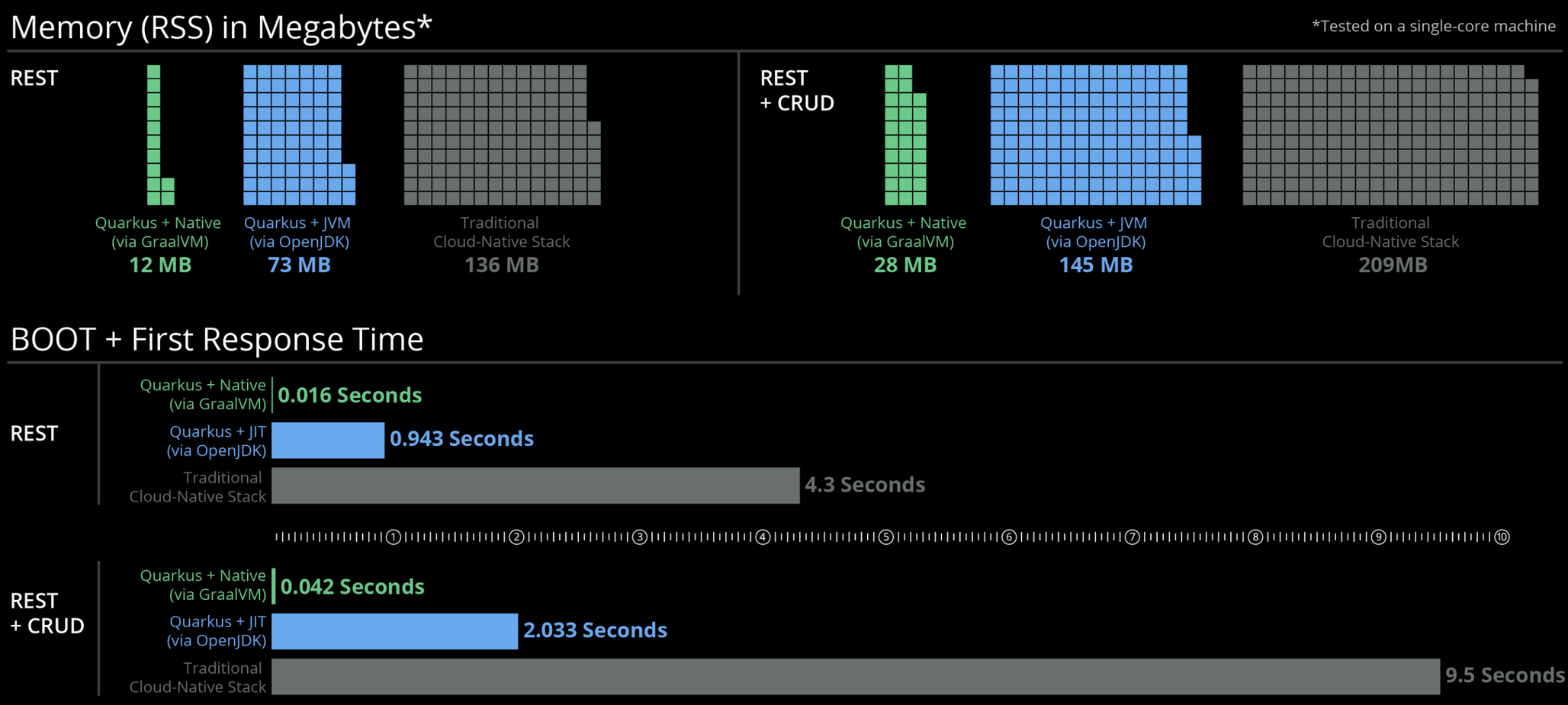
Why Quarkus?

- Container first
- Unifies imperative and reactive
- Developer joy

Container First

- Optimized for low memory and fast startup:
- First class support for Graal/SubstrateVM
- Build-time metadata processing
- Reduction in use of reflection
- Native image pre-boot

Container First



Unifies Imperative and Reactive

- The client/server model has changed
- Quarkus supports today's paradigms:
  - HTTP microservices
  - reactive applications
  - message-driven microservices
  - serverless

Imperative

```
@Inject
SayService say;

@GET
@Produces(MediaType.TEXT_PLAIN)
public String hello() {
  return say.hello();
  }
```

Reactive

```
@Inject @Channel("kafka")
Publisher<String> reactiveSay;

@GET
@Produces(MediaType.SERVER_SENT_EVENTS)
public Publisher<String> stream() {
  return reactiveSay;
  }
```

Developer Joy

- Unified configuration
- Zero configuration
- Streamlined code (80/20)
- No-hassle native executable generation

Best-of-Breed Libraries

- Quarkus leverages a multitude of libraries

Best-of-Breed Libraries

- Eclipse MicroProfile
- Eclipse Vert.x
- JAX-RS
- RESTEasy
- CDI
- Netty
- Hibernate
- JPA
- JTA
- Apache Kafka
- Apache Camel

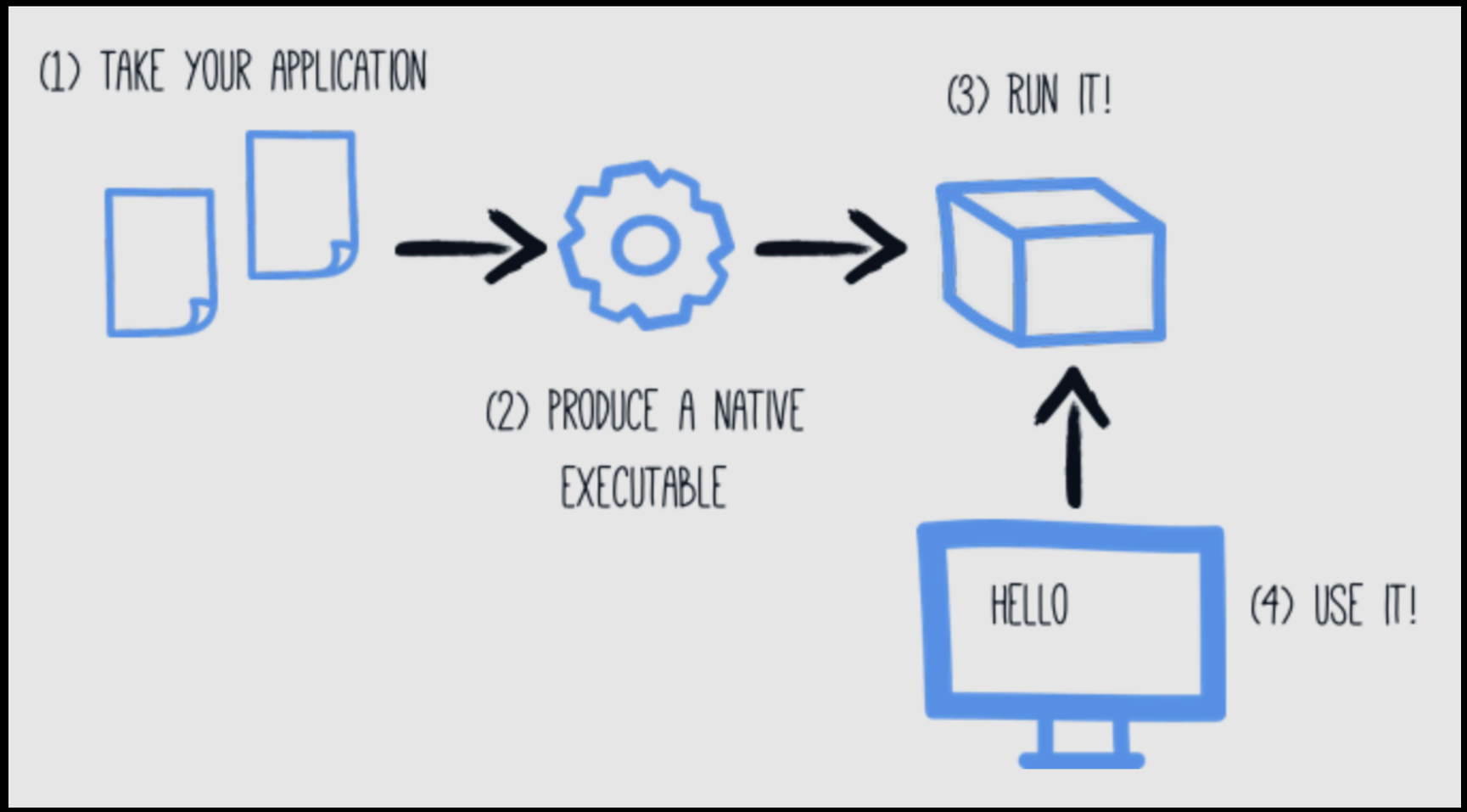
Let's Get Started…

Default Project

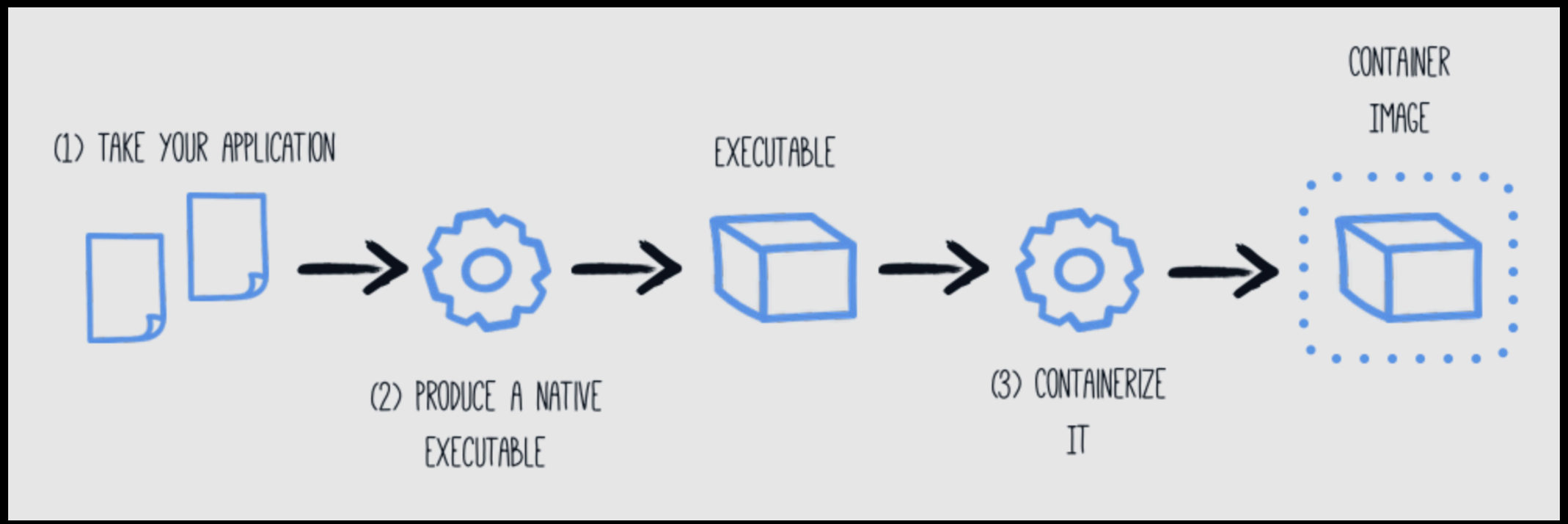
```
$ mvn io.quarkus:quarkus-maven-plugin:1.1.0.Final:create \
    -DprojectGroupId=org.acme \
    -DprojectArtifactId=getting-started \
    -DclassName="org.acme.quickstart.GreetingResource" \
    -Dpath="/hello"

$ cd getting-started

$ mvn compile quarks:dev
```

Hello

Native Image



Container


(1) TAKE YOUR APPLICATION
EXECUTABLE
CONTAINER
IMAGE
(2) PRODUCE A NATIVE
EXECUTABLE
(3) CONTAINERIZE
IT

Quarkus Resources

- `https://quarkus.io`
- `https://infoq.com/news/2019/03/redhat-release-quarkus/`
- `https://infoq.com/articles/redhat-release-quarkus-1-0/`

Contact Info

mike@redlich.net

@mpredli

redlich.net/portfolio

github.com/mpredli01

Thanks!